Technical Report
97-12

Proof Polynomials vs. λ-terms

S. N. ARTEMOV

December 1997

REPORT DOCUMENTATION PAGE	Form Approved OMB NO. 0704-0188

1. AGENCY USE ONLY (Leave Blank)	2. REPORT DATE **31 March 1998**	3. REPORT TYPE AND DATES COVERED **Technical Report**

4. TITLE AND SUBTITLE **Proof Polynomials vs. λ–terms**	5. FUNDING NUMBERS **DAAH04-96-1-0341**
6. AUTHOR(S) **S. N. Artemov**	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Regents of the University of California** **c/o Sponsored Projects Office** **336 Sproul Hall** **Berkeley, CA 94720-5940**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) U. S. Army Research Office P.O. Box 12211 Research Triangle Park, NC 27709-2211	10. SPONSORING / MONITORING AGENCY REPORT NUMBER ARO 35873.72-MA-MUR

11. SUPPLEMENTARY NOTES
The views, opinions and/or findings contained in this report are those of the author(s) and should not be construed as an official Department of the Army position, policy or decision, unless so designated by the documentation.

12 a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution unlimited.	12 b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

The Logic of Proofs provides a basic framework for the formalization of reasoning about proofs. It incorporates proof terms into the propositional language, using labeled logical operators "t:" with the intended reading of t:F being "T is a proof of F". In the current paper we demonstrate how the typed . λ–calculus and the modal λ–calculus can be realized in the Logic of Proofs.

19980519 186

14. SUBJECT TERMS **logic of proofs, intuitionistic logic, automated deductions**	15. NUMBER OF PAGES **18**
	16. PRICE CODE

17. SECURITY CLASSIFICATION OR REPORT UNCLASSIFIED	18. SECURITY CLASSIFICATION ON THIS PAGE UNCLASSIFIED	19. SECURITY CLASSIFICATION OF ABSTRACT UNCLASSIFIED	20. LIMITATION OF ABSTRACT UL

NSN 7540-01-280-5500

Standard Form 298 (Rev.2-89)
Prescribed by ANSI Std. 239-18
298-102

Technical Report
97-12

Proof Polynomials vs. λ-terms

S. N. ARTEMOV

December 1997

Proof Polynomials vs. λ-terms

Sergei N. Artemov [*]

Abstract

The Logic of Proofs ($\mathcal{LP}$) introduced in [2] provides a basic framework for the formalization of reasoning about proofs. It incorporates proof terms into the propositional language, using labeled logical operators "$t:$" with the intended reading of $t:F$ being "*t is a proof of F*". $\mathcal{LP}$ is supplied with an exact provability semantics in Peano Arithmetic, a simple axiom system, and completeness and decidability theorems. $\mathcal{LP}$ naturally expresses a number of constructions of logic involving the notion of proof, which have previously been formulated and/or interpreted in an informal metalanguage, e.g. modal logic, Intuitionistic logic with its Brouwer-Heyting-Kolmogorov semantics, etc. ([2], [3]). In the current paper we demonstrate how the typed λ-calculus and the modal λ-calculus can be realized in the Logic of Proofs.

1 Introduction

The Logic of Proofs ($\mathcal{LP}$) incorporates proof terms directly into the propositional language using new logical operators t: labeled by special proof terms with the intended reading of $t:F$ being "*t is a proof of F*" (cf. [2]). Three basic operations on proofs are postulated: *application, proof checker*, and *choice*. The language of $\mathcal{LP}$ has an exact intended semantics, where "t is a proof of F" is interpreted as a corresponding arithmetical formula of provability in Peano Arithmetic $\mathcal{PA}$ about the codes of t and F (cf. Section 4). A natural axiom system for $\mathcal{LP}$ along with the completeness theorem of this axiom system with respect to the arithmetical semantics was found in [2].

The intuitionistic logic $\mathcal{I}nt$ has an informal Brouwer-Heyting-Kolmogorov (*BHK*) operational semantics ([9], [10], cf. [18], [7], [19]) given in terms of logical conditions on the formulas and their proofs. A well-known formalization of the *BHK* operations is made in the Curry-Howard presentation of intuitionistic deductions as typed lambda terms, leading to the "*Propositions as Types*" paradigm. This duality now plays a key role in some fields of proof theory, automated deductions and the logical theory of computation. (Cf. [13] and [15] for an impressive current list of applications of proof motivated λ-calculi).

[*]Center for Foundations of Intelligent Systems, 625 Rhodes Hall, Cornell University, Ithaca NY, 14853 `artemov@hybrid.cornell.edu`. Research supported by the ARO under the MURI program "Integrated Approach to Intelligent Systems", grant number DAA H04-96-1-0341.

In the current paper we show how to represent the typed λ-calculi in $\mathcal{LP}$ directly. Under this embedding, the formation rules for the λ-terms become admissible rules of $\mathcal{LP}$. In fact, the λ-calculus for $\mathcal{I}nt$ can be defined in the Horn Logic of Proofs $\mathcal{HLP}$, which is the fragment of the Intuitionistic Logic of Proofs with Horn formulas only, without "proof checker", "choice" operations, or nesting of proof terms. The Intuitionistic Logic of Proofs $\mathcal{ILP}$ has a natural provability semantics with respect to Heyting Arithmetic $\mathcal{HA}$; $\mathcal{ILP}$ is also a natural dynamic counterpart of the intuitionistic modal logic $\mathbf{IS4}_{\Box}$ (cf. [4], [13], [15]). All these give a provability semantics for the typed λ-calculus and for the modal λ-calculus ([13], [15]).

2 Logic of Proofs

The language of $\mathcal{LP}$ contains

sentence variables $p_0, \ldots, p_n, \ldots$, boolean constants $\top, \bot$,
proof variables $x_0, \ldots, x_n, \ldots$
boolean connectives $\rightarrow, \ldots$
functional symbols: monadic !, binary + and $\cdot$
operator symbol of the type *term*:*formula*.

Terms and formulas are defined in the natural way: proof variables are terms; sentence variables and boolean constants are formulas; whenever s, t are terms $!t, (s+t), (s\cdot t)$ are again terms. Boolean connectives behave conventionally, and for t a term and F a formula $t:F$ is a formula. We will write st instead of $(s\cdot t)$ and omit parentheses when convenient. If $\vec{x} = (x_1, \ldots, x_n)$ and $\Gamma = (A_1, \ldots, A_n)$, then we will write $\vec{x}:\Gamma$ for $x_1:A_1, \ldots, x_n:A_n$. The intended semantics of this language is:

proof term = finite set of proofs = nondeterministic proof,

$t:F$ = "*t contains a proof for F*".

The basic operations on proof terms are interpreted as *application* ($\cdot$), *proof checker* (!), and *choice* (+). The meaning of these operations is specified in the axiom system for $\mathcal{LP}$ below and in a precise arithmetical provability semantics in [2] (cf. Section 4). Under this arithmetical interpretation "*t contains a proof for F*" is represented by a Godel provability formula in $\mathcal{PA}$. The use of the arithmetic is not essential here. $\mathcal{LP}$ is taylored to describe any system which is able to argue about its own proofs.

2.1 Definition. The system $\mathcal{LP}$.

Axioms:

Cl. Axioms of classical propositional logic in the language of $\mathcal{LP}$

A1. $t{:}F \to F$ *"verification principle"*

A2. $t{:}(F \to G) \to (s{:}F \to (t{\cdot}s){:}G)$ *"application"*

A3. $t{:}F \to\ !t{:}(t{:}F)$ *"proof checker"*

A4. $s{:}F \to (s{+}t){:}F, \quad t{:}F \to (s{+}t){:}F$ *"choice"*

Rules of inference:

$$\frac{F \quad F \to G}{G}$$ *"modus ponens"*

$$\frac{F}{t{:}F}$$ *for any formula F and proof term t* *"necessitation"*

The *Necessitation* rule reflects the *formalization principle*: "a given proof can be formalized and put into any given finite set of formal proofs".

The derivations from hypothesises in $\mathcal{LP}$ are defined in a usual way with the following convention: the *necessitation* rule can be used only if F is derived in $\mathcal{LP}$ without any hypothesises.

A *Terms Specification* (*TS*) is a finite set of formulas $t_1 : F_1, \ldots, t_n : F_n$ provable in $\mathcal{LP}$. Each *TS* may be considered as a partial specification of proof terms $t_1 ;, \ldots, t_n :$ as proofs for the formulas $F_1, \ldots, F_n$ respecively. Each derivation in $\mathcal{LP}$ naturally generates a terms specification $t_1 : F_1, \ldots, t_n : F_n$, consisting of all formulas $t_i : F_i$ introduced in this derivation by the *necessitation* rule.

For the usual Gödel proof predicate $Proof(x, y)$ in $\mathcal{PA}$ which formalizes the relation

"x is a code of a derivation of a formula with a code y".

there are primitive recursive functions from codes of proofs to codes of proofs corresponding to "$\cdot$" and "!": "$\cdot$" stands for a operation on proof sequences which realizes the *modus ponens* rule in arithmetic, and "!" is the "proof checker" operation, appearing in the proof of the second Gödel Incompleteness theorem (cf. [17], [5]). However, the *choice* operation "+" is already incompatible with the deterministic character of $Proof(x, y)$, where a proof x proves only one formula y. Indeed, if $s : F$ and $t : G$, then both $(s + t) : F$ and $(s + t) : G$, i.e. $s + t$ proves at least two different formulas F and G.

The usual proof predicate has a natural nondeterministic version $PROOF(x, y)$ called *the standard nondeterministic proof predicate*

"x is a code of a derivation **containing** *a formula with a code y"*.

PROOF already has all three of the operations of the $\mathcal{LP}$-language: the operation $s+t$ is now just a concatenation of (nondeterministic) proofs s and t.

The "+"-free fragment $\mathcal{LP}^-$ of the logic of proofs deserves separate attention as a *neutral logic of proofs.* $\mathcal{LP}^-$ does not specify the determinacy of a proof predicate. $\mathcal{LP}^-$ can by expanded to the operational logic of functional proofs ($\mathcal{FLP}$) developed in [12] by adding a special *functionality* axiom which by means of unification captures on the propositional level the deterministic character of a proof predicate (cf. the system $\mathcal{F}$ from [1]).

$$\begin{tikzcd}
\textit{Logic of (nondeterministic) proofs } \mathcal{LP} & & \textit{Logic of deterministic proofs } \mathcal{FLP} \\
& \textit{Neutral logic of proofs } \mathcal{LP}^- \arrow[ul] \arrow[ur] &
\end{tikzcd}$$

An arithmetical completeness theorem for the logic of proofs ([2], cf. 4.1) holds for $\mathcal{LP}^-$ as well, the proof remains essentially intact. However, the operation "+" is needed to realize the entire modal logic and the modal λ-calculus. In this paper we consider $\mathcal{LP}$ as the basic system for the logic of proofs.

2.2 Comment. No single operator "t:" in $\mathcal{LP}$ is not a normal modality since none of them satisfies the property $t:(p\to q)\to(t:p\to t:q)$ for propositional variables p and q. The usual Kripke semantics for modal logics does not work for the Logic of Proofs. These make $\mathcal{LP}$ fundamentally different from numerous multimodal logics, e.g. the dynamic logic of programs ([11]), where the modality is upgraded by some additional features. In turn, in the Logic of Proofs the modality is **decomposed** into a family of proof terms generated by the operations "application", "proof checker", and "choice": $\mathcal{S}4$ is a forgetful projection of $\mathcal{LP}$.

$$\begin{tikzcd}
& & \textit{polymodal logics} \\
& & \cdots \\
\textit{Logic of Proofs} & \textit{Modal Logic} \arrow[l, "\text{decomposition}"'] \arrow[uur] \arrow[r, "\text{upgrading}"] \arrow[ddr] & \textit{dynamic logic of programs} \\
& & \cdots \\
& & \textit{logics of knowledge}
\end{tikzcd}$$

This decomposition appeared to be fruitful in revealing the provability content of the classical modal logic $\mathcal{S}4$ ([2], cf. also sections 3 and 4 of this paper). Similarly, every structure containing the $\mathcal{S}4$-type modality, e.g. in the modal λ-calculus (cf. Section 6), may be decomposed by means of $\mathcal{LP}$-terms.

2.3 Comment. The usual deduction theorem holds for $\mathcal{LP}$:

$$\Gamma, A \vdash_{\mathcal{LP}} B \;\;\Rightarrow\;\; \Gamma \vdash_{\mathcal{LP}} A \rightarrow B.$$

The standard proof of the deduction theorem remains intact with the new clause when B is introduced by *necessitation*:

$$\frac{\Gamma, A \vdash_{\mathcal{LP}} B}{\Gamma, A \vdash_{\mathcal{LP}} t{:}B}.$$

By the convention on the use of *necessitation*, $\vdash_{\mathcal{LP}} t{:}B$ and thus $\Gamma \vdash_{\mathcal{LP}} A \rightarrow t{:}B$. Note, that the deduction theorem provides a linear time[1] algorithm which given a derivation $\Gamma, A \vdash_{\mathcal{LP}} B$ constructs a derivation $\Gamma \vdash_{\mathcal{LP}} A \rightarrow B$.

2.4 Lemma. (Substitution lemma for $\mathcal{LP}$). *If $\Gamma(x,p) \vdash_{\mathcal{LP}} B(x,p)$ for a propositional variable p and a proof variable x, then for any proof term t and any formula F*

$$\Gamma(x/t, p/F) \vdash_{\mathcal{LP}} B(x/t, p/F).$$

Proof. All axioms and rules of $\mathcal{LP}$ remain axioms and rules after a substitution.
◀

2.5 Lemma. *The following rules are admissible in $\mathcal{LP}$. Here A,B are $\mathcal{LP}$-formulas, Γ, Δ are finite sets of $\mathcal{LP}$-formulas, y is a proof variable, t is a proof term, $\vec{y}$ and $\vec{s}$ are vectors of proof variables and proof terms respectively, "$\vdash$" means "$\vdash_{\mathcal{LP}}$".*

Lifting:

$$\frac{\vec{s}{:}\Gamma, \Delta \vdash B}{\vec{s}{:}\Gamma, \vec{y}{:}\Delta \vdash t(\vec{y}){:}B} \qquad \textit{for some } t(\vec{y});$$

[1]With a fast access to Γ, A.

Lowering: $$\frac{\Gamma, \vec{y}{:}\Delta \vdash t{:}B}{\Gamma, \Delta \vdash B};$$
(provided $\vec{y}$ does not occur in the conclusion)

Abstraction: $$\frac{\vec{s}{:}\Gamma, y{:}A \vdash t(y){:}B}{\vec{s}{:}\Gamma \vdash \lambda y.t(y){:}(A \to B)}$$
for some proof term denoted by $\lambda y.t(y)$
(provided y does not occur in the conclusion)

Proof. *Lifting.* By induction on a proof of B from the premises $\vec{s}{:}\Gamma, \Delta$. If $B \in \vec{s}{:}\Gamma$, then $\vec{s}:\Gamma, \vec{y}:\Delta \vdash !s_i : B$ for some $s_i \in \vec{s}$. If $B \in \Delta$, then $y_j : B \in \vec{y} : \Delta$ for some $y_j \in \vec{y}$. If B is an axiom, then $t : B$ may be derived by the *necessitation* rule. Let B be obtained from C, $C \to B$ by *modus ponens.* Then, by the induction hypothesis, $\vec{s}:\Gamma, \vec{y}:\Delta \vdash t_1(\vec{y}) : (C \to B)$ and $\vec{s}:\Gamma, \vec{y}:\Delta \vdash t_2(\vec{y}) : C$. for some terms t_1 and t_2. By A2, $\vec{x}:\Gamma, \vec{y}:\Delta \vdash (t_1 \cdot t_2) : B$. Let $B = s:C$ be derived by *necessitation* from C. Then $\vdash s:C$ and thus $\vdash t:(s:C)$, by one more use of *necessitation.* This proof gives a quadratic algorithm of constructing a proof $\vec{s}{:}\Gamma, \vec{y}{:}\Delta \vdash t(\vec{y}){:}B$ from a proof $\vec{s}{:}\Gamma, \Delta \vdash B$.

Lowering. From $\Gamma, \vec{y}:\Delta \vdash t:B$ conclude $\Gamma, \vec{y}:\Delta \vdash B$. Note that none of the variables from $\vec{y} = (y_1, \ldots, y_n)$ occurs in Γ, Δ, B. Define an operation $'$ on $\mathcal{LP}$-formulas: $p = p'$ for a propositional variable p, $'$ commutes with boolean connectives and

$$(s{:}F)' = \begin{cases} F', & \text{if } s \text{ contains a variable from } \vec{y} \\ s{:}(F'), & \text{otherwise.} \end{cases}$$

By a straightforward induction on the derivation length show that for each F from the derivation $\Gamma, \vec{y}{:}\Delta \vdash B$

$$\begin{cases} \text{if } \Gamma, \vec{y}{:}\Delta \vdash F \text{ then } \Gamma, \Delta \vdash F' \\ \text{if } \vdash F \text{ then } \vdash F'. \end{cases}$$

In particular, $\Gamma, \Delta \vdash B$.

Case 1. F is from $\Gamma, \vec{y}{:}\Delta$. Easy, since $\Gamma' = \Gamma$ and $(\vec{y}{:}\Delta)' = \Delta$.
Case 2. F is a propositional axiom. Then F' is the same axiom.
Case 3. $F = s{:}X \to X$.
 a) s is $\vec{y}$-free. Then $F' = s{:}X' \to X'$, an axiom A1.
 b) s is not $\vec{y}$-free. Then $F' = X' \to X'$.
Case 4. $F = s{:}(X \to Y) \to (r{:}X \to lsrY)$.
 a) s, r are both $\vec{y}$-free. Then F' is again an axiom A2.
 b) s is $\vec{y}$-free, r is not. Then $F' = s{:}(X' \to Y') \to (X' \to Y')$, axiom A1.
 c) r is $\vec{y}$-free, s is not. Then $F' = (X' \to Y') \to (r{:}X' \to Y')$, derivable in $\mathcal{LP}$ since $r{:}X' \to Y'$.

d) s, r are both not $\vec{y}$-free. Then $F' = (X' \rightarrow Y') \rightarrow (X' \rightarrow Y')$.

Case 5. $F = s:X \rightarrow !s:(s:X)$.

a) s is $\vec{y}$-free. Then F' is again an axiom A3.

b) s is not $\vec{y}$-free. Then $F' = X' \rightarrow X'$.

Case 6. $F = s:X \ \rightarrow \ (s+r):X$.

a) s, r are both $\vec{y}$-free. Then F' is again an axiom A4.

b) s is $\vec{y}$-free, r is not. Then $F' = s:X' \rightarrow X'$, axiom A1.

c) s is not $\vec{y}$-free. Then $F' = X' \rightarrow X'$.

Case 7. $F = t:X \ \rightarrow \ (s+r):X$. Similar to Case 6.

Case 8. F is obtained from $X, X \rightarrow F$ by *modus ponens*. Then F' is obtained from $G', X' \rightarrow X'$ by the same rule.

Case 9. $F = s:X$ is obtained by *necessitation* from X. Then $\vdash_{\mathcal{LP}} X$ and $\vdash_{\mathcal{LP}} s:X$. By the Inductiopn hypothesis, $\vdash_{\mathcal{LP}} X'$.

a) s is $\vec{y}$-free. Then $F' = s:X'$ and $\vdash_{\mathcal{LP}} F'$ is obtained from $\vdash_{\mathcal{LP}} X'$ by *necessitation.*

b) s is not $\vec{y}$-free. Then $F' = X'$.

Note that this proof delivers a linear time algorithm of transforming a derivation $\Gamma, \vec{y}:\Delta \vdash t:B$ into a derivation $\Gamma, \Delta \vdash B$.

Abstraction. By Lowering, from $\vec{s}:\Gamma, y:A \vdash t(y):B$ get $\vec{s}:\Gamma, A \vdash B$. By Deduction, get $\vec{s}:\Gamma \vdash A \rightarrow B$, and by Lifting, get $\vec{s}:\Gamma \vdash r:(A \rightarrow B)$ for some proof term r.

◀

2.6 Comment. The term $t(\vec{y})$ introduced by the Lifting rule is nothing but a protocol for a proof of B from $\vec{s}:\Gamma, \vec{y}:\Delta$. The same holds for the rule of Abstraction, where $\lambda y.t(y)$ is a protocol for a proof of $A \rightarrow B$ from $\vec{s}:\Gamma$.

The Abstraction rule might not look like an operation on terms, because in the the process of constructing $\lambda y.t(y)$ from $t(y)$ we get rid of the latter and seemingly construct $\lambda y.t(y)$ from the scratch. However, this is not the case. A term $t(y)$ is a protocol of a derivation of B from $\vec{s}:\Gamma, y:A$. From this derivation we get a derivation $\vec{s}:\Gamma, A \vdash B$, then a derivation of $A \rightarrow B$ from $\vec{s}:\Gamma$. Finally, $\lambda y.t(y)$ is a protocol for the latter derivation. The proof of 2.5 gives a quadratic algorithm which transforms a derivation $\vec{s}:\Gamma, y:A \vdash t(y):B$ into a derivation $\vec{s}:\Gamma \vdash \lambda y.t(y):(A \rightarrow B)$.

3 Realization of $\mathcal{S}4$ in $\mathcal{LP}$.

3.1 Example. $\mathcal{S}4 \vdash (\Box A \wedge \Box B) \rightarrow \Box(A \wedge B)$.

In $\mathcal{LP}$ this can be reproduced by the following:

1. $A, B \vdash A \wedge B$

2. $x{:}A, y{:}B \vdash t(x,y){:}(A\wedge B)$, from 1. by Lifting
3. $x{:}A \wedge y{:}B \vdash t(x,y){:}(A\wedge B)$
4. $\vdash (x{:}A \wedge y{:}B) \rightarrow t(x,y){:}(A\wedge B)$

In fact, here $t(x,y)$ can be taken $(cx)y$, where $c{:}(A \rightarrow (B \rightarrow (A\wedge B)))$ is introduced by the *necessitation* rule.

3.2 Example. $\mathcal{S}4 \vdash (\Box A \vee \Box B) \rightarrow \Box(A\vee B)$.

In $\mathcal{LP}$ the corresponding derivation is

1. $A \vdash A\vee B$
2. $B \vdash A\vee B$
3. $x{:}A \vdash (a{\cdot}x){:}(A\vee B)$, where $a{:}(A \rightarrow A\vee B)$ is obtained by *necessitation*,
4. $y{:}B \vdash (b{\cdot}y){:}(A\vee B)$, where $b{:}(B \rightarrow A\vee B)$ is obtained by *necessitation*,
5. $x{:}A \vdash (ax{+}by){:}(A\vee B)$, $y{:}B \vdash (ax{+}by){:}(A\vee B)$ by A4 from 3, 4.
6. $x{:}A \vee y{:}B \vdash (ax{+}by){:}(A\vee B)$
7. $\vdash (x{:}A \vee y{:}B) \rightarrow (ax{+}by){:}(A\vee B)$

In fact **all** $\mathcal{S}4$-theorems have a corresponding operational reading in $\mathcal{LP}$.

3.3 Definition. By an *$\mathcal{LP}$-realization* r of a modal formula F we mean an assignment of $\mathcal{LP}$-terms to all occurrences of the modality in F.

Let F^r denote the image of F under a realization r. Positive and negative occurrences of modal operators in a formula and a sequent are defined in the usual way. A realization r is *normal* if all negative occurrences of $\Box$ are realized by proof variables.

3.4 Theorem. ([2]) *If $\mathcal{S}4 \vdash F$, then $\mathcal{LP} \vdash F^r$ for some normal realization r.*

The proof describes a quadratic algorithm which for a given cut-free derivation $\mathcal{T}$ in $\mathcal{S}4$ assigns $\mathcal{LP}$ terms to all occurrences of the modality in $\mathcal{T}$.

4 Arithmetical Semantics

Let us agree to use a new functional symbol $\iota z\varphi(z)$ for each arithmetical formula $\varphi(z)$ and assume that ι-terms could be eliminated in the usual way by using the small scope convention (cf. [6]). An arithmetical formula φ is *provably* Δ_1 iff both φ and $\neg\varphi$ are provably Σ_1. A term $\iota z\varphi$ is *provably recursive* iff φ is provably Σ_1. *A closed recursive term* is a provably total, and a provably recursive term $\iota z\varphi$ such that φ contains no free variables other than z. Close recursive terms are our provably recursive names for natural numbers. We have to use all of them as proof realizers, since some operations on proofs, e.g. the *proof checker*

"!", depend on the name of the argument, not on its value. Indeed, if $PROOF(\overline{n},\overline{k})$ holds, then $PROOF(\overline{n}+0,\overline{k})$ also holds, $!(\overline{n})$ is a proof of $PROOF(\overline{n},\overline{k})$ and $!(\overline{n}+0)$ is a proof of $PROOF(\overline{n}+0,\overline{k})$. However, $!(\overline{n})$ and $!(\overline{n}+0)$ deliver proofs of different formulas, thus, generally speaking, $!(\overline{n}) \neq !(\overline{n}+0)$.

A *proof predicate* is a provably Δ_1-formula $Prf(x,y)$ such that for all φ

$$\mathcal{PA} \vdash \varphi \quad \Leftrightarrow \quad \text{for some } n \in \omega \quad Prf(n, \ulcorner\varphi\urcorner) \text{ holds.}$$

A proof predicate *Prf(x,y)* is *normal* if

1) for every proof k the set $T(k) = \{l \mid Prf(k,l)\}$ is finite and the function

$$\widetilde{T(k)} = \textit{the code of}\, T(k)$$

is provably recursive,

2) for every finite set S of theorems of $\mathcal{PA}$, $S \subseteq T(k)$ for some proof k.

The nondeterministic proof predicate *PROOF* (above) is an example of a normal proof predicate.

For every normal proof predicate *Prf* there are provably recursive terms $m(x,y)$, $a(x,y)$, $c(x)$ such that for all closed recursive terms s,t and for all arithmetical formulas φ,ψ the following formulas are valid:

$$Prf(s,\ulcorner\varphi\to\psi\urcorner) \wedge Prf(t,\ulcorner\varphi\urcorner) \to Prf(m(s,t),\ulcorner\psi\urcorner)$$

$$Prf(s,\ulcorner\varphi\urcorner) \to Prf(a(s,t),\ulcorner\varphi\urcorner), \qquad Prf(t,\ulcorner\varphi\urcorner) \to Prf(a(s,t),\ulcorner\varphi\urcorner)$$

$$Prf(t,\ulcorner\varphi\urcorner) \to Prf(c(\ulcorner t\urcorner),\ulcorner Prf(t,\ulcorner\varphi\urcorner)\urcorner).$$

4.1 Definition. An arithmetical *interpretation* $*$ of $\mathcal{LP}$-language has the following parameters:

- a normal proof predicate *Prf*,
- an evaluation of sentence letters by sentences of arithmetic,
- and an evaluation of proof letters by closed recursive terms.

We put $\top^* \equiv (0=0)$ and $\bot^* \equiv (0=1)$. $*$ commutes with boolean connectives, $(t\cdot s)^* \equiv m(t^*,s^*)$, $(t+s)^* \equiv a(t^*,s^*)$, $(!t)^* \equiv c(\ulcorner t^*\urcorner)$, $(t{:}F)^* \equiv Prf(t^*,\ulcorner F^*\urcorner)$. Under any interpretation $*$ an $\mathcal{LP}$-term t becomes a closed recursive term t^* (i.e. a recursive name of a natural number), and an $\mathcal{LP}$-formula F becomes an arithmetical sentence F^*.

Let *TS* be a terms specification. An arithmetical interpretation $*$ is *TS-interpretation* if $\mathcal{PA} \vdash G^*$ for all $G \in TS$.

In what follows "arithmetically TS-valid" means either "provable in $\mathcal{PA}$ under any TS-interpretation" or "true in the standard model of arithmetic under any TS-interpretation".

Note that the reflexivity principle for a modal logic becomes valid in the proof semantics, since $t{:}F \to F$ is provable in $\mathcal{PA}$ under any interpretation $*$. Indeed, let n be the value of t^*. If $Prf(n, \ulcorner F^* \urcorner)$ is true, then $\mathcal{PA} \vdash F^*$, thus $\mathcal{PA} \vdash Prf(n, \ulcorner F^* \urcorner) \to F^*$. If $Prf(n, \ulcorner F^* \urcorner)$ is false, then $\mathcal{PA} \vdash \neg Prf(n, \ulcorner F^* \urcorner)$, and again $\mathcal{PA} \vdash Prf(n, \ulcorner F^* \urcorner) \to F^*$.

4.2 Theorem. ([2], Arithmetical completeness of $\mathcal{LP}$) *Let TS be an arbitrary terms specification, then*

$$\mathcal{LP} \vdash F \textit{ with a terms specification } TS \quad \Leftrightarrow \quad F^* \textit{ is arithmetically } TS\textit{-valid.}$$

Combining 3.4 and 4.1, we obtain the arithmetical completeness of $\mathcal{S}4$:

$$\mathcal{S}4 \vdash F \quad \Leftrightarrow \quad F^r \textit{ is arithmetically } TS\textit{-valid for some (normal) realization } r \textit{ and some terms specification } TS.$$

Gödel in [8] defined a translation tr of intuitionistic formulas, into $\mathcal{S}4$-formulas where $tr(F)$ is obtained from F by boxing all atoms and all implications in F. This Gödel translation is shown ([8], [14]) to provide a faithful embedding of $\mathcal{I}nt$ in $\mathcal{S}4$. The proof interpretation of $\mathcal{LP}$-terms above provides a faithful proof arithmetical realization of $\mathcal{I}nt$:

$$\mathcal{I}nt \vdash F \quad \Leftrightarrow \quad [tr(F)]^r \textit{ is arithmetically } TS\textit{-valid for some (normal) realization } r \textit{ and some terms specification } TS.$$

A direct realization of $\mathcal{I}nt$ in $\mathcal{LP}$ is presented in [3].

5 On the Intuitionistic and Horn Logic of Proofs

5.1 Definition. The *Intuitionistic Logic of Proofs*, $\mathcal{ILP}$, is a version of $\mathcal{LP}$ (Definition 2.1) with the propositional axioms for the intuitionistic logic $\mathcal{I}nt$, instead of Cl.

5.2 Definition. A *Horn formula* of the language of $\mathcal{LP}$ is a disjunction C of literals of the form $t{:}F$, where t is a proof term with the operations *application* only, F is a plain propositional formula and C has at most one positive literal. The Horn formulas are presented below as sequents $\Gamma \Rightarrow A$, with Γ a (possibly empty) set of positive literals, and A a positive literal. The *Horn Logic of Proofs* $\mathcal{HLP}$ consists of the Horn formulas provable in $\mathcal{ILP}$.

An inspection of the corresponding proofs shows that the straightforward "intuitionistic" versions of Lemmas 2.4, 2.5 (admissible rules in the Logic of Proofs) hold with $\mathcal{ILP}$ and $\mathcal{HLP}$ instead of $\mathcal{LP}$.

Under $\mathbf{IS4}_\Box$ we mean an intuitionistic modal logic, introduced in [4] (cf. also [13], [15]). Theorem 3.4 (realization of modal logic), holds with $\mathcal{ILP}$ instead of $\mathcal{LP}$ and $\mathbf{IS4}_\Box$ instead of $\mathcal{S}4$. Therefore, $\mathcal{ILP}$ is a dynamic version of $\mathbf{IS4}_\Box$ in the same sense as $\mathcal{LP}$ is a dynamic version of $\mathcal{S}4$.

Modal λ-calculi for $\mathbf{IS4}_\Box$ have been discribed in [13], [15], where one can also find an impressive list of possible applications. As we will observe later, $\mathcal{HLP}$ and $\mathcal{ILP}$ naturally emulate the typed λ-calculus for the intuitionistic logic and the modal λ-calculus for $\mathbf{IS4}_\Box$ respectively. Thus $\mathcal{ILP}$ naturally contains both: $\mathbf{IS4}_\Box$ and the modal λ-calculus as its term-forgetful projections.

Also $\mathcal{ILP}$ has an intended arithmetical interpretation over Heyting Arithmetic $\mathcal{HA}$ in the style of Section 4: "$t\!:\!F$" is interpreted as an arithmetical formula

$$Proof_{\mathcal{HA}}(t^*, \ulcorner F^* \urcorner),$$

saying that "t^* is a proof of F^* in $\mathcal{HA}$". Thus both $\mathcal{ILP}$ and $\mathcal{HLP}$ enjoy an intuitionistic arithmetical provability semantics, and supply a typed λ-calculus for intuitionistic logic and a modal λ-calculus for $\mathbf{IS4}_\Box$ with arithmetical provability semantics.

6 Logic of Proofs *vs* Lambda Calculi.

Now we show how to realize Curry-Howard λ-terms for $\mathcal{I}nt$ in $\mathcal{HLP}$, modal λ-terms for $\mathbf{IS4}_\Box$ in $\mathcal{ILP}$ (and thus both of these λ-calculi in $\mathcal{LP}$). The basic ingradients for that, including the rule of Abstraction are all present in Lemma 2.5.

6.1 Theorem. *The following are admissible rules in $\mathcal{LP}$, in $\mathcal{ILP}$ and in $\mathcal{HLP}$ for the corresponding* $\mathbf{p}_0$, $\mathbf{p}_1$, $\mathbf{p}$, $\mathbf{k}_0$, $\mathbf{k}_1$, $E^{\vee}_{x,y}$ *(x, y, u, u_0, u_1 are proof variables, w is a fresh variable, u does not occur in t):*

$$\frac{u_i\!:\!A_i, \Gamma \vdash t\!:\!C}{\{w\!:\!(A_0 \wedge A_1), \Gamma \vdash t\!:\!C\}\sigma} \qquad \frac{\Gamma \vdash t\!:\!A \quad \Gamma \vdash s\!:\!B}{\Gamma \vdash \mathbf{p}(t,s)\!:\!(A \wedge B)}$$

(σ is the substitution $[u_i/\mathbf{p}_i w]$)

$$\frac{x\!:\!A, \vec{s}\!:\!\Gamma \vdash t_0\!:\!C \quad y\!:\!B, \vec{s}\!:\!\Gamma \vdash t_1\!:\!C}{w\!:\!(A \vee B), \vec{s}\!:\!\Gamma \vdash E^{\vee}_{x,y}(w,t_0,t_1)\!:\!C} \qquad \frac{\Gamma \vdash t\!:\!A_i}{\Gamma \vdash \mathbf{k}_i\!:\!(A_0 \vee A_1)}$$

$$\frac{\Gamma \vdash t\!:\!A \quad u\!:\!B, \Gamma \vdash s\!:\!C}{\{w\!:\!(A \to B), \Gamma \vdash s\!:\!C\}\sigma} \qquad \frac{u\!:\!A, \vec{s}\!:\!\Gamma \vdash t\!:\!B}{\vec{s}\!:\!\Gamma \vdash \lambda u.t\!:\!(A \to B)}$$

(σ is the substitution $[u/w\cdot t]$)

Proof. In $(\wedge, \vdash)$ rule we have to find a proof term for $\mathbf{p}_i$. Consider the following chain of derivations in $\mathcal{LP}$:

1. $u_i\!:\!A_i, \Gamma \vdash t(u_i)\!:\!C$, by the induction hypothesis,

2. $\Gamma \vdash u_i\!:\!A_i \to t(u_i)\!:\!C$, by the deduction from 1,
3. $\vdash a_i\!:\!(A_0 \wedge A_1 \to A_i)$, by *necessitation*,
4. $w\!:\!(A_0 \wedge A_1) \vdash a_i w\!:\!A_i$, by application from 3,
5. $w\!:\!(A_0\sigma \wedge A_1\sigma) \vdash a_i w\!:\!A_i\sigma$, by $\sigma = a_i w$ from 4,
6. $a_i w\!:\!A_i\sigma, \Gamma\sigma \vdash t(a_i w)\!:\!C\sigma$, from 1,
7. $w\!:\!(A_0\sigma \wedge A_1\sigma), \Gamma\sigma \vdash t(a_i w)\!:\!C\sigma$, from 5, 6,

and then we may put $\mathbf{p}_i = a_i$.

For the rule $(\vdash,\wedge)$ we have to realize $\mathbf{p}(t,s)$:

1. $\Gamma \vdash t\!:\!A, s\!:\!B$, from the premises,
2. $\vdash a\!:\!(A \to (B \to (A \wedge B)))$, by *necessitation*,
3. $t\!:\!A, s\!:\!B \vdash (as)t\!:\!(A \wedge B)$, from 2,
4. $\Gamma \vdash (as)t\!:\!(A \wedge B)$, from 1, 3.

Now we put $\mathbf{p}(u,v) = (a \cdot v) \cdot u$.

For $(\vee, \vdash)$ we have to evaluate $E^{\vee}_{x,y}(w,t_0,t_1)$, which is a term built from w, t_0, t_1.

1. $\vec{s}\!:\!\Gamma \vdash \lambda x.t_0(x)\!:\!(A \to C)$; $\vec{s}\!:\!\Gamma \vdash \lambda y.t_1(y)\!:\!(B \to C)$, by Abstraction from 1,
2. $\vdash a\!:\!((A \to C) \to ((B \to C) \to (A \vee B \to C)))$, by *necessitation*,
3. $\lambda x.t_0(x)\!:\!(A \to C), \lambda y.t_1(y)\!:\!(B \to C), w\!:\!(A \vee B) \vdash (a \cdot \lambda x.t_0 \cdot \lambda y.t_1 \cdot w)\!:\!C$, by application from 2,
4. $\vec{s}\!:\!\Gamma, w\!:\!(A \vee B) \vdash (a \cdot \lambda x.t_0 \cdot \lambda y.t_1 \cdot w)\!:\!C$, from 1 and 3,

and we put $E^{\vee}_{x,y}(w,t_0,t_1)$ to be $a \cdot \lambda x.t_0 \cdot \lambda y.t_1 \cdot w$;

In the rule $(\vdash, \vee)$ we have to find $\mathbf{k}_i$.

1. $\Gamma \vdash t\!:\!A$, from the premise,
2. $\vdash b_i\!:\!(A_i \to (A_0 \vee A_1))$, by *necessitation*,
3. $t\!:\!A \vdash (b_i \cdot t)\!:\!(A_0 \vee A_1)$, from 2,
4. $\Gamma \vdash (b_i \cdot t)\!:\!(A_0 \vee A_1)$, from 1, 3.

Put $\mathbf{k}_i$ to be $b_i \cdot t$.

The admissibility of the rule $(\to, \Rightarrow)$:

1. $\Gamma, t\!:\!A \to u\!:\!B \vdash s\!:\!C$, from the premises,
2. $\vdash w\!:\!(A\sigma \to B\sigma) \to (t\!:\!A\sigma \to wt\!:\!B\sigma)$, by *necessitation*,
3. $\Gamma\sigma, t\!:\!A\sigma \to wt\!:\!B\sigma \vdash (s\!:\!C)\sigma$, by substitution σ from 1,
5. $\Gamma\sigma, w\!:\!(A\sigma \to B\sigma) \vdash (s\!:\!C)\sigma$, from 4, 5.

At last, the rule $(\vdash, \to)$ is a special case of the Abstraction rule for $\mathcal{LP}$.
◀

6.2 Corollary. *The λ-calculus for Int can be realized in the Horn Logic of Proofs so that*

$x_1{:}A_1,\ldots,x_n{:}A_n \Rightarrow t{:}B$ *is derivable in the λ-calculus*	$\Longrightarrow$	$x_1{:}A_1,\ldots,x_n{:}A_n \vdash t^*{:}B$ *is an admissible rule of $\mathcal{HLP}$ for some propof term t^*.*

Proof. Take a "sequent style" formulation of the λ calculus, e.g. **G2i*** from [19] and use a straightforward induction on the derivations and theorem 6.1.
◀

6.3 Corollary. *The modal λ-calculus can be realized in the Intuitionistic Logic of Proofs:*

$x_1{:}A_1,\ldots,x_n{:}A_n \Rightarrow t(\vec{x}){:}B$ *is derivable in the modal λ-calculus*	$\Longrightarrow$	$x_1{:}A_1^r,\ldots,x_n{:}A_n^r \vdash_{\mathcal{ILP}} t^*(\vec{x}){:}B^r$. *for some realization r of the modal language in $\mathcal{LP}$ and some proof term t^**

Proof. As above all the usual steps of λ-terms construction can be emulated in the Logic of Proofs (here in $\mathcal{ILP}$). The new "modal" operations on λ-terms: "*box*" and "*unbox*" are naturally represented by *Lifting* and *Lowering* (cf. 2.5) respectively. However, in the modal λ-calculus the types are the plain modal formulas, and in the Logic of Proofs the formulas are dynamic. So, in order to apply the admissible rules from 2.5 and 6.1 to stipulate the process of λ-terms constructing we have to agree the languages of $\mathbf{IS4}_\Box$ and $\mathcal{ILP}$ by realizing $\mathbf{IS4}_\Box$ in $\mathcal{ILP}$ according to Section 5. Now the chain of trasformations leading from a modal λ-term $t(\vec{x})$ to the corresponding proof term $t^*(\vec{x})$ is the following:

1. Take a modal λ-term in a full form with premises and types

$$x_1{:}A_1,\ldots,x_n{:}A_n \Rightarrow t(\vec{x}){:}B,$$

where $A_1,\ldots,A_n,B$ are plain modal formulas.

2. Consider the corresponding $\mathbf{IS4}_\Box$-derivation

$$A_1,\ldots,A_n \vdash_{\mathbf{IS4}_\Box} B.$$

3. Present this derivation in a dynamic form by the algorithm realizing $\mathbf{IS4}_\Box$ into $\mathcal{ILP}$, i.e. assign proof terms to all the occurrences of the modalities in the $\mathbf{IS4}_\Box$-derivation above to get a similar derivation in $\mathcal{ILP}$:

$$A_1^r,\ldots,A_n^r \vdash_{\mathcal{ILP}} B^r.$$

Without loss of generality we assume, that the variables for the realization r are all different from the ones used in $t(\vec{x})$.

4. Repeat the formation steps for $t(\vec{x})$ from 1 in the $\mathcal{ILP}$-language using 2.5 and 6.1 to get the desired

$$x_1 : A_1^r, \ldots, x_n : A_n^r \vdash_{\mathcal{ILP}} t^*(\vec{x}) : B^r.$$

◀

6.4 Example. Let us compare two realizations of the $\mathcal{ILP}$-theorem $(\Box A \wedge \Box B) \rightarrow \Box(A \wedge B)$ by a modal λ-term and by an $\mathcal{LP}$-term. To make things more transparent we will work with natural style derivations. The proof in $\mathbf{IS4}_\Box$ is

$$\dfrac{\Box A \vee \Box B \qquad \dfrac{\{\Box A\} \qquad \dfrac{\dfrac{\{\Box A\}}{A}}{A \vee B}}{\Box(A \vee B)} \qquad \dfrac{\{\Box B\} \qquad \dfrac{\dfrac{\{\Box B\}}{B}}{A \vee B}}{\Box(A \vee B)}}{\Box(A \vee B)},$$

where the brackets { } denote the discharged premises. A corresponding modal λ-term in the notations of [4] is constructed as follows

$$\dfrac{z : (\Box A \vee \Box B) \qquad \dfrac{\{x : \Box A\} \qquad \dfrac{\dfrac{\{x : \Box A\}}{\mathtt{unbox}(x) : A}}{\mathtt{inr}\cdot\mathtt{unbox}(x) : A \vee B}}{\mathtt{box}\cdot\mathtt{inr}\cdot\mathtt{unbox}(x) : \Box(A \vee B)} \qquad \dfrac{\{y : \Box B\} \qquad \dfrac{\dfrac{\{y : \Box B\}}{\mathtt{unbox}(y) : B}}{\mathtt{inl}\cdot\mathtt{unbox}(y) : A \vee B}}{\mathtt{box}\cdot\mathtt{inl}\cdot\mathtt{unbox}(y) : \Box(A \vee B)}}{\mathtt{case}\ z\ \mathtt{of}\ \mathtt{inl}(x)\ \mathtt{then}\ \mathtt{box}\cdot\mathtt{inl}\cdot\mathtt{unbox}(x) \| \mathtt{inr}(y)\ \mathtt{then}\ \mathtt{box}\cdot\mathtt{inl}\cdot\mathtt{unbox}(y) : \Box(A \vee B)},$$

The corresponding $\mathcal{ILP}$-proof in the notations $[\![t]\!]F$ for $t : F$ is

$$\dfrac{[\![u]\!]A \vee [\![v]\!]B \qquad \dfrac{\{[\![u]\!]A\} \qquad \dfrac{\dfrac{\{[\![u]\!]A\}}{A}}{A \vee B}}{[\![au]\!](A \vee B)} \qquad \dfrac{\{[\![v]\!]B\} \qquad \dfrac{\dfrac{\{[\![v]\!]B\}}{B}}{A \vee B}}{[\![bv]\!](A \vee B)}}{[\![au + bv]\!](A \vee B)},$$

with the axiom constants specification: $[\![a]\!](A \rightarrow (A \vee B))$ and $[\![b]\!](B \rightarrow (A \vee B))$.

To construct an $\mathcal{ILP}$-realization we do not need to evaluate all the entries of the proof:

$$\frac{z:(\llbracket u\rrbracket A\vee\llbracket v\rrbracket B)\qquad \dfrac{x:\{\llbracket u\rrbracket A\}\quad \dfrac{\dfrac{\{\llbracket u\rrbracket A\}}{A}}{A\vee B}}{l_0(x):\llbracket au\rrbracket(A\vee B)}\qquad \dfrac{y:\{\llbracket v\rrbracket B\}\quad \dfrac{\dfrac{\{\llbracket v\rrbracket B\}}{B}}{A\vee B}}{l_1(y):\llbracket bv\rrbracket(A\vee B)}}{(d\cdot\lambda x.l_0(x)\cdot\lambda y.l_1(y)\cdot z):\llbracket au+bv\rrbracket(A\vee B)}.$$

Here $l_0(x)$ is the Lifting $\mathcal{LP}$-term from

$$\frac{\llbracket u\rrbracket A\vdash\llbracket au\rrbracket(A\vee B)}{x:\llbracket u\rrbracket A\vdash l_0(x):\llbracket au\rrbracket(A\vee B)},$$

$l_1(y)$ is the Lifting term from

$$\frac{\llbracket v\rrbracket B\vdash\llbracket bv\rrbracket(A\vee B)}{y:\llbracket v\rrbracket B\vdash l_1(y):\llbracket bv\rrbracket(A\vee B)},$$

and d is specified as

$$d:((X\to Z)\to((Y\to Z)\to((X\vee Y)\to Z))),$$

where X is $\llbracket u\rrbracket A$, Y is $\llbracket v\rrbracket B$ and Z is $\llbracket au+bv\rrbracket(A\vee B)$.

7 Conclusions

1. The Logic of Proofs is a very simple extension of the propositional logic by proof terms generated by only three operations: unary *proof checker*, and binary *application* and *choice*. These operations along with the entire family of proof terms have an exact intended provability semantics in arithmetic. In fact *proof checker* and *application* first appeared implicitly in the Second Gödel Incompleteness Theorem, and the Logic of Proofs discloses a fundamental connection of this theorem with the classical modal logic $\mathcal{S}4$, the Intuitionistic logic and the λ-calculi.

2. Some basic logical notions can be naturally emulated in the Logic of Proofs, e.g. *modality* and *application/λ-abstraction.* As the result, $\mathcal{LP}$ contains $\mathcal{S}4$, the intuitionistic $\mathcal{S}4$, and modal λ-calculus as special "term-forgetful" projections.

$$\begin{tikzcd}
 & & \textit{Logic of Proofs} & & \\
 & & & \textit{Intuitionistic Logic of Proofs} \arrow[ul] & \\
\mathcal{S}4 \arrow[uurr] & & \textit{Modal } \lambda\textit{-calculus} \arrow[ur] & & \textit{Horn Logic of Proofs} \arrow[ul] \\
 & \textit{Intuitionistic } \mathcal{S}4 \arrow[ul] \arrow[ur] & & \lambda\textit{-calculus} \arrow[ul] \arrow[ur] & \\
 & & \textit{Intuitionistic Logic} \arrow[ul] \arrow[ur] & &
\end{tikzcd}$$

All the usual λ-terms constructors may be assumed to be present in $\mathcal{LP}$ explicitly.

3. Proof terms from $\mathcal{LP}$ essentially enrich the languages of both the modal logic and the modal λ-calculus. Proof terms are polymorphic[2]. A type in $\mathcal{LP}$ (i.e. an $\mathcal{LP}$-formula) may contain proof terms of any type, including its own.

4. The operations *application, proof checker* and *choice* are characteristics of the language of logic, not a particular proof system; the arithmetical semantics for $\mathcal{LP}$ covers all recursive self-referential systems of proofs, not just natural deductions in propositional logics. According to Curry-Howard,

$$\lambda\textit{-terms} = \textit{natural deduction proof protocols.}$$

The Logic of Proofs extends this proof semantics to:

$$\textit{proof terms} = \textit{all proof protocols with self-referential capacities.}$$

Thus the Logic of Proofs places λ-calculi in a general provability context.

5. The computational content of the λ-terms is also preserved, since $\mathcal{LP}$ allows a computational reading of proof terms. In addition, $\mathcal{LP}$ contains definable λ-terms for all classical derivations thus providing a framework for reasoning about constructive and classical proofs together, about relatively computable λ-terms, etc.

6. From the technical point of view, $\mathcal{LP}$ gives system independent sufficient conditions for a logic/theory to contain definable λ-terms. To represent the usual λ-calculus it suffices to have "application of proof terms" operation only, "proofs" of certain propositional axioms,

[2]In the Functional Logic of Proofs $\mathcal{FLP}$ ([12]) a proof term t has an exact type (a formula, proven by t).

no matter in what system, and to enjoy some trivial closure properties, like the deduction theorem.

Acknowledgements.

I am greatly indebted to Anil Nerode for his constant encouragement and support. It is a pleasure to acknowledge also valuable discussions with Lev Beklemishev, Robert Constable, Vladimir Krupski, Grisha Mints, Rohit Parikh, and Andre Scedrov. My thanks to Todd Wilson for helping me with references to some recent papers on modal λ-calculi.

The research described in this publication was supported in part by the Russian Foundation for Basic Research, grant 96-01-01395.

References

[1] S. Artëmov, "Logic of Proofs," *Annals of Pure and Applied Logic*, v. 67 (1994), pp. 29-59.

[2] S. Artëmov, "Operational Modal Logic," *Tech. Rep. MSI 95-29*, Cornell University, December 1995.

[3] S. Artëmov. "Proof Realization of Intuitionistic and Modal Logics", *Tech. Rep. MSI 96-06, Cornell University*, December 1996

[4] G. Bierman and V. de Paiva, "Intuitionistic necessity revisited.", *Proceedings of the Logic at Work Conference,* Amsterdam (December 1992), Second revision, June 1996 (http://theory.doc.ic.ac.uk/tfm/papers.html).

[5] G. Boolos, *Logic of Provability.*, CUP, 1993.

[6] D. van Dalen, *Logic and Structure*, Springer-Verlag, 1994.

[7] J.-Y. Girard, Y. Lafont and P. Taylor, *Proofs and Types*, Cambridge University Press, 1989.

[8] K. Gödel, "Eine Interpretation des intuitionistischen Aussagenkalkuls", *Ergebnisse Math. Colloq.*, Bd. 4 (1933), S. 39-40.

[9] A. Heyting, "Die intuitionistische Grundlegung der Mathematik", *Erkenntnis*, Bd. 2 (1931), S. 106-115.

[10] A. Kolmogoroff, "Zur Deutung der intuitionistischen Logik," *Math. Ztschr.*, Bd. 35 (1932), S.58-65.

[11] D. Kozen and J. Tiuryn, "Logic of Programs", in *Handbook of Theoretical Computer Science. Volume B, Formal Models and Semantics*, The MIT Press/Elsevier, pp. 789-840, 1990

[12] V.N. Krupski, "Operational Logic of Proofs with Functionality Condition on Proof Predicate", Lecture Notes in Computer Science, v. 1234, *Logical Foundations of Computer Science' 97, Yaroslavl'*, pp. 167-177, 1997

[13] S. Martini and A. Masini, "A computational interpretation of modal proofs", in Wansing, ed., *Proof Theory of Modal Logics*, (Workshop proceedings), Kluwer, 1994.

[14] J.C.C. McKinsey and A. Tarski, "Some theorems about the sentential calculi of Lewis and Heyting", Journal of Symbolic Logic, v. 13 (1948), pp. 1-15.

[15] F. Pfenning and H.C. Wong, "On a modal lambda-calculus for S4", *Electronic Notes in Computer Science* 1, 1995.

[16] T. Sidon, "Provability Logic with Operations on Proofs", Lecture Notes in Computer Science, v. 1234, *Logical Foundations of Computer Science' 97, Yaroslavl'*, pp. 342-353, 1997

[17] C. Smorynski, "The incompleteness theorems", in *Handbook of mathematical logic*, Amsterdam; North Holland, 1977, pp. 821-865.

[18] A.S. Troelstra and D. van Dalen, *Constructivism in Mathematics. An Introduction*, v. 1, Amsterdam; North Holland, 1988.

[19] A.S. Troelstra and H. Schwichtenberg, *Basic Proof Theory*, Cambridge University Press, 1996.